מועד הבחינה:
אביב תשפ"ב – 2022 – מועד ב'
מספר השאלון: 97105
נספח ממשקים לבחינה JAVA
נספח ממשקים לבחינה #C

מבני נתונים ותכנות מונחה עצמים

הנדסאים וטכנאים – הנדסת תוכנה

הנחיות לבחינה

א. משך הבחינה: ארבע שעות וחצי.

ב. מבנה השאלון ומפתח ההערכה:

בשאלון זה שני מבחנים, <u>עליכם לענות על מבחן אחד בלבד בהתאם למוסד הלימודים</u>:

מבחן ב- Java (עמוד 2)

מבחן ב- #C (עמוד 17)

בכל מבחן 11 שאלות.

חלק א' – 45 נקודות

שאלות 1-4: יש לענות על **שלוש** שאלות בלבד. ערך כל שאלה 15 נקודות.

חלק ב' – 30 נקודות

שאלות 5-8: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 15 נקודות.

חלק ג' – 25 נקודות

שאלות 9-11: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 12 נקודות.

נקודה אחת תינתן על הערכה.

בסך הכול: 100 נקודות.

ג. חומר עזר מותר לשימוש:

1. מחשבון (אין להשתמש במחשב כף יד או במחשבון עם תקשורת חיצונית).
2. קלסר **<u>אחד בלבד</u>** עם **<u>חומר ההרצאות</u>**. אין להוציא דפים מהקלסר.

אין לצרף ספרים או חוברות עם פתרונות.

ד. הוראות כלליות:

1. יש לקרוא בעיון את ההנחיות בדף השער ואת כל שאלות הבחינה, ולוודא שהן מובנות.
2. את התשובות יש לכתוב בצורה מסודרת, בכתב יד ברור ונקי (גם בכך תלויה הערכת הבחינה).
3. יש להשאיר את העמוד הראשון במחברת הבחינה ריק. בסיום המבחן יש לרשום בעמוד זה את מספרי התשובות לבדיקה. התשובות ייבדקו לפי סדר כתיבתן בעמוד זה. לא ייבדקו תשובות עודפות.
4. יש לכתוב את התשובות במחברת הבחינה **בעט בלבד**, בכתב יד ברור.
5. יש להתחיל כל תשובה בעמוד חדש ולציין את מספר השאלה ואת הסעיף. אין צורך להעתיק את השאלה עצמה.
6. טיוטה יש לכתוב במחברת הבחינה בלבד. יש לרשום את המילה "טיוטה" בראש העמוד ולהעביר עליו קו כדי שלא ייבדק.
7. יש להציג פתרון מלא ומנומק, כולל חישובים לפי הצורך. הצגת תשובה סופית ללא שלבי הפתרון לא תזכה בניקוד.
8. יש להסביר בפירוט כל תוכנית שנכתבה, **תוכנית ללא הסבר מפורט לא תזכה בניקוד**.
9. אם לדעתכם חסר בשאלה נתון, יש לציין זאת ולהוסיף נתון מתאים שיאפשר לכם להמשיך בפתרון השאלה, נמקו את בחירתכם.

חל איסור מוחלט להוציא שאלון או מחברת בחינה מחדר הבחינה!

בהצלחה!

בשאלון זה 31 עמודי בחינה ו- 4 עמודי נספחים.

מבחן ב- JAVA

חלק א׳

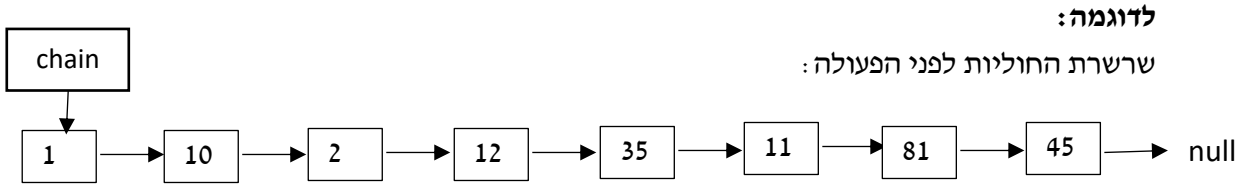
ענו על <u>שלוש</u> מבין השאלות 1-4 (ערך כל שאלה – 15 נקודות).

שאלה 1

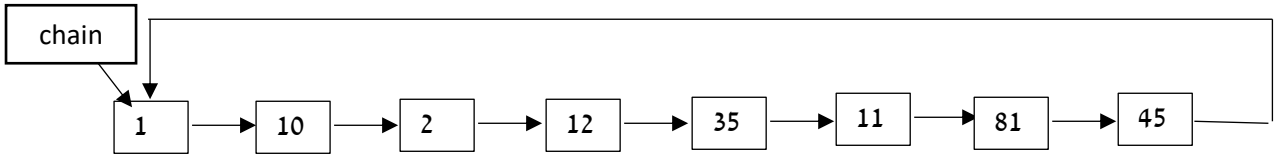
(8 נק׳) א. כתבו פעולה המקבלת הפנייה לחוליה ראשונה בשרשרת חוליות של מספרים שלמים והופכת את השרשרת לשרשרת ״**מעגלית**״, כך שהחוליה אחרי החוליה האחרונה תהיה שוב החוליה הראשונה.

לדוגמה:

שרשרת החוליות לפני הפעולה:



שרשרת מעגלית אחרי הפעולה



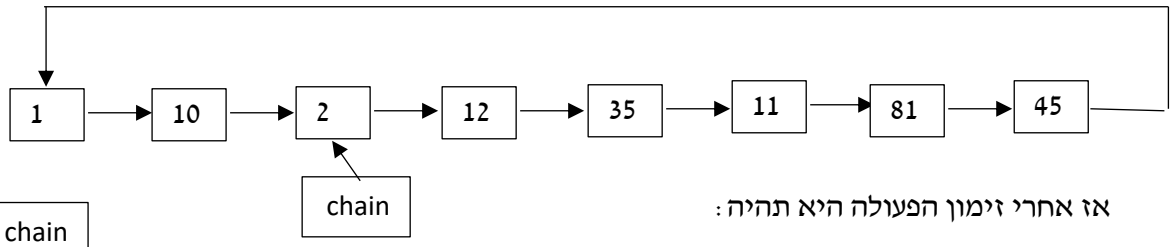
כותרת הפעולה:

```
public static void toCircle(Node<Integer> chain)
```

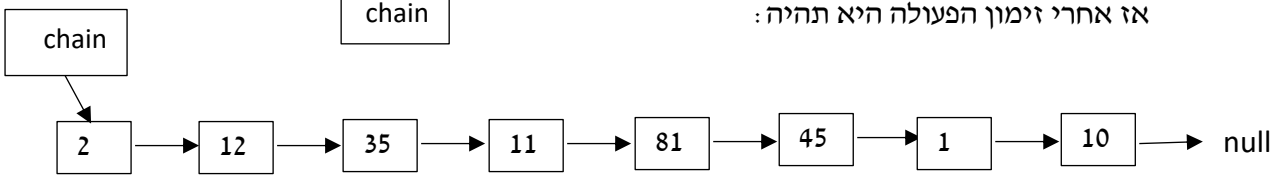
(7 נק׳) ב. כתבו פעולה המקבלת הפנייה לחוליה כלשהי בשרשרת חוליות **מעגלית** של מספרים שלמים והופכת אותה לשרשרת לינארית, כך שהחוליה שהפנייה אליה התקבלה כפרמטר, תהיה חוליה ראשונה של השרשרת והחוליה שהייתה לפניה תהיה החוליה האחרונה בשרשרת.

לדוגמה:

אם לפני זימון הפעולה השרשרת הייתה:



אז אחרי זימון הפעולה היא תהיה:

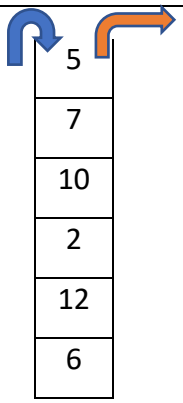
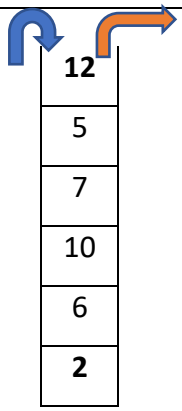


כותרת הפעולה: `public static void toLine(Node<Integer> chain)`

שאלה 2

(12 נק׳) א. כתבו פעולה המקבלת מחסנית של מספרים שלמים. הפעולה תעביר את הערך הקטן ביותר לתחתית המחסנית ואת הערך הגדול ביותר לראש המחסנית.
אפשר להניח שהערך הקטן ביותר וגם הערך הגדול ביותר מופיעים במחסנית פעם אחת בלבד.

לדוגמה:

המחסנית לפני הפעולה	**המחסנית אחרי הפעולה**
	

(3 נק׳) ב. מהי הסיבוכיות של הפעולה שכתבתם בסעיף א׳? הסבירו את תשובתכם.

שאלה 3

נתונות שתי המחלקות הבאות:

```
public class One  {
    private int num1=0;
    private int num2=0;
    protected static int count =0;

    public One(int num)    {
        this.num1 = num;
        this.num2 = num;
        count++;
        System.out.println ("One ctor1");
    }

    public One(int num1, int num2)    {
        this.num1 = num1;
        this.num2 = num2;
        count++;
        System.out.println ("One ctor2");
    }

    public int sum()    {return this.num1+this.num2;}
    public void setNum(int num)    { this.num2 = num;}
    public static int getCount()    { return count;    }
}// end of class One

public class Two extends One  {
    private int num3 = 0;

    public Two(int num)  {
        super (num);
        this.num3 = num;
        System.out.println ("Two ctor1");
    }

    public Two(int num1, int num2, int num3)  {
        super(num1, num2);
        this.num3 = num3;
        System.out.println ("Two ctor2");
    }

    public int sum()  {return super.sum()+this.num3; }
    public void setNum(int num)   {this.num3 = num; }
    public static int getCount()  {return count;    }
}// end of class Two
```

נתונה המחלקה Driver הבאה:

```
public class Run  {
    public static void main (String [] args)     {
            One f1 = new One(10);
            One f2 = new One (10, 20);
            Two s1 = new Two(1);
            One f3 = new Two(2);
(1)         System.out.println ("count= " + One.getCount());
            f2 = s1;
(2)         System.out.println ("sum= " + f2.sum());
            s1.setNum(2);
(3)         System.out.println ("sum= " + s1.sum());
(4)         System.out.println ("sum= " + f2.sum());
            f2.setNum(4);
(5)         System.out.println ("sum= " + f2.sum());
            f1.setNum(4);
(6)         System.out.println ("sum= " + f1.sum());
(7)         System.out.println ("count= " + Two.getCount());
    }
}
```

מהו הפלט המתקבל לאחר הפעלת השיטה main שבמחלקה Run?

כתבו את מספר השורה ואז את הפלט המתקבל משורה זו. אם יש שגיאת קומפילציה, כתבו אותה.

שאלה 4

נתונות שלוש מחלקות A ,B ,C הבאות:

```
public class A {
    public static int count = 0;
    private int num;
    public A()
    {
        count++;
        num = count;
    }
    public A(int n)
    {
        num = n;
    }
    public String toString()
    {
        return count+","+ num;
    }
} // end of A
public class B extends A {
    private String str;
    public B()
    {
        str="GOOD";
    }
    public B(String s)
    {
        super(10);
        str = s;
    }
    public String toString()
    {
        return super.toString()+" "+str;
    }
}// end of B
public class C extends B {
    private String str;
    public C(String s)
    {
        str = s;
    }
    public String toString()
    {
        return super.toString()+" "+str;
    }// end of C
```

(6 נק׳) א. נתונה מחלקה Driver הבאה:

```
public class Driver
{
    public static void main(String[] args)
    {
        A[] arr=new A[6];

        arr[0]= … // (1)
        arr[1]= … // (2)
        arr[2]= … // (3)
        arr[3]= … // (4)
        arr[4]= … // (5)
        arr[5]= … // (6)


        for(int i=0; i<arr.length; i++)
            System.out.println(arr[i]);
    }
}
```

השלימו פעולות הסרות (1)-(6) כך שהפלט יהיה:

```
4,10 Morning

4,1

4,2 GOOD Afternoon

4,10 Evening

4,3 GOOD Night

4,4 GOOD
```

(6 נק׳) ב. הכותרת של המחלקה C השתנתה ל-

`public class C extends A`

האם שינוי זה יגרום לשגיאה? אם כן – הסבירו את השגיאה, אם לא – רשמו מה יהיה הפלט של ה- main אחרי השינוי.

(3 נק׳) ג. הכותרת של המחלקה C השתנתה ל-

`public class C`

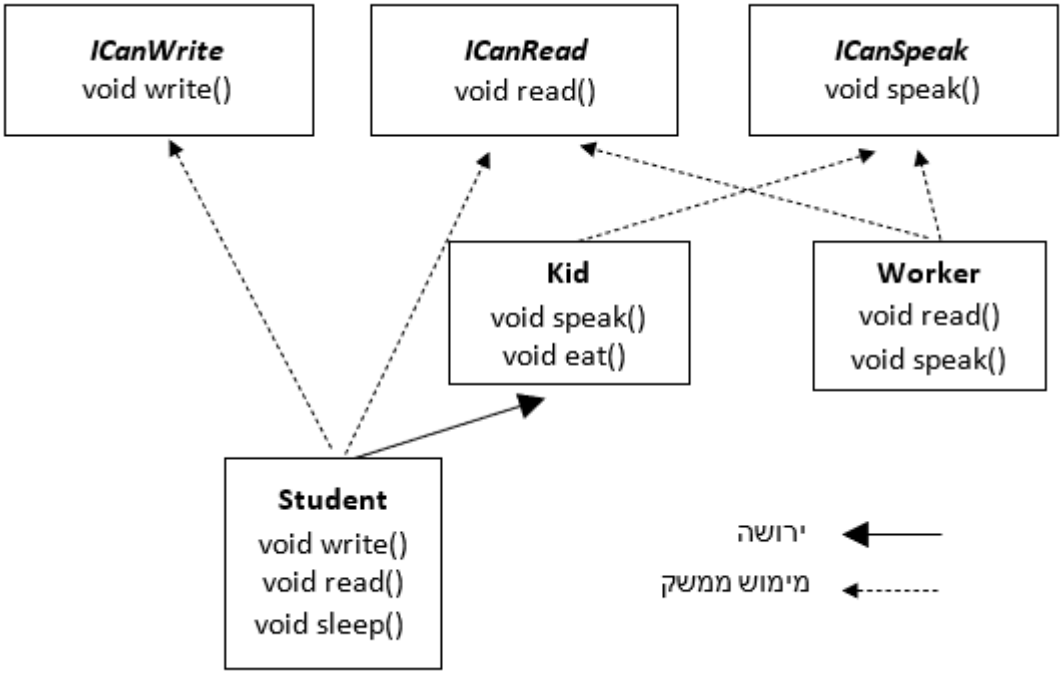
האם שינוי זה יגרום לשגיאה? אם כן – הסבירו את השגיאה, אם לא – רשמו מה יהיה הפלט של ה- main אחרי השינוי.

חלק ב׳

ענו על <u>שתיים</u> מבין השאלות 5-8 (ערך כל שאלה – 15 נקודות).

שאלה 5

נתונה היררכיית המחלקות הבאה הכוללת שלוש מחלקות Kid, Student, Worker ושלושה ממשקים ICanRead, ICanWrite, ICanSpeak:



בכל מחלקה מוגדרת פעולה בונה ללא פרמטרים.

(3 נק׳) א. כתבו עבור כל מחלקה, את כותרת המחלקה.

(6 נק׳) ב. השלימו את שלוש שורות (1)-(3) כך שקטע קוד שלפניכם יעבור בלי שגיאות קומפילציה וזמן ריצה:

```
_____(1)_____   x = new Worker();
_____(2)_____   y = new Kid();
_____(3)_____   z = new Student();
x = y;
y = new Student();
y.eat();
z.read();
((Student)z).speak();
((Kid)z).eat();
z = new Worker();
```

(6 נק׳) ג. כתבו פעולה המקבלת כפרמטר מערך מטיפוס ICanSpeak. הפעולה תעבור על איברי המערך ותקרא לפעולה ()write, אם היא קיימת בעבור העצם. אם הפעולה ()write אינה קיימת – תיקרא הפעולה ()speak.

שאלה 6

שיטה אחת לדחיסת טקסט היא להחליף רצף של תווים זהים בתו אחד.

כדי לממש את השיטה הוגדרה המחלקה הבאה:

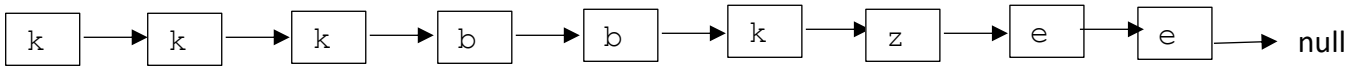
```
public class CharInt
{
    private char let;
    private int num;
    public CharInt(char let, int num)
    {
        this.let = let;
        this.num = num;
    }
  ….
}
```

תכונות המחלקה הן: תו (let) ומספר פעמים שהוא מופיע ברצף (num). במחלקה הוגדרו פעולות set/get לכל תכונה וגם פעולה toString.

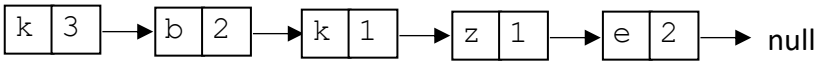
(9 נק׳) א. כתבו פעולה המקבלת הפניה לחוליה הראשונה של שרשרת חוליות של תווים (״טקסט מקורי״).
הפעולה מחזירה את הפניה לחוליה הראשונה של שרשרת חדשה של עצמים מטיפוס CharInt (״טקסט דחוס״).

לדוגמה:

אם שרשרת התווים היא:



אז הפעולה תחזיר את השרשרת הבאה:



כותרת הפעולה:

```
public static Node<CharInt> zip(Node<Character> chain)
```

(6 נק׳) ב. כתבו פעולה המקבלת הפניה לחוליה הראשונה של שרשרת המייצגת ״טקסט דחוס״ ומחזירה את הפניה לחוליה הראשונה של שרשרת תווים המייצגת ״טקסט מקורי״.

כותרת הפעולה:

```
public static Node<Character> unZip(Node<CharInt> chain)
```

שאלה 7

נגדיר טיפוס נתונים חדש בשם **DoubleQueue (תור כפול)**, כמבנה המכיל שני תורים של מספרים שלמים:

```
class DoubleQueue {
    private Queue<Integer> first;
    private Queue<Integer> second;
```

במחלקה הוגדרו ארבע פעולות הבאות **(אין צורך לממש אותן)**:

- הפעולה `public int count(int queueID)` מחזירה מספר איברים בתור `queueID`.
- הפעולה `public void addValue (int num,int queueID)` מוסיפה את הערך `num` לתור `queueID`.

(4 נק') א. כתבו פעולה המקבלת מספר תור `queueID` ומחזירה ערך הגדול ביותר בתור.

כותרת הפעולה:

```
public int maxValue(int queueID)
```

(4 נק') ב. כתבו פעולה המקבלת מספר שלם `num` ומספר תור `queueID`.

כותרת הפעולה:

```
public void eraseValue(int num, int queueID)
```

הפעולה מוחקת מתור `queueID` את המספר `num`. אם המספר `num` מופיע יותר מפעם אחת בתור, הפעולה תמחק רק מופע אחד של המספר.

אם המספר `num` לא מופיע בתור, הפעולה אינה מבצעת דבר.

(7 נק') ג. כתבו את הפעולה

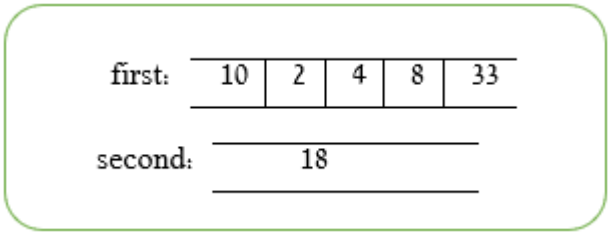
```
public static void order(DoubleQueue dq)
```

הפעולה מקבלת תור כפול של מספרים שלמים dq. מספר האיברים בתור dq <u>זוגי</u>.

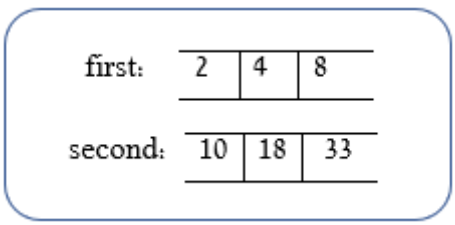
הפעולה תחלק את האיברים בין התורים first ו- second, לפי הכלל הבא:

- מספר האיברים בשני התורים זהה.
- ההפרש בין **סכום** האיברים הנמצאים בתור second ו**סכום** האיברים הנמצאים בתור first יהיה הגדול ביותר.

לדוגמה: תור כפול לפני זימון הפעולה order:



תור כפול אחרי זימון הפעולה order:



שימו לב! בסעיף ג' אפשר להשתמש רק בפעולות שהוגדרו במחלקה DoubleQueue.

אין להוסיף פעולות נוספות או להשתמש במבני נתונים נוספים!

שאלה 8

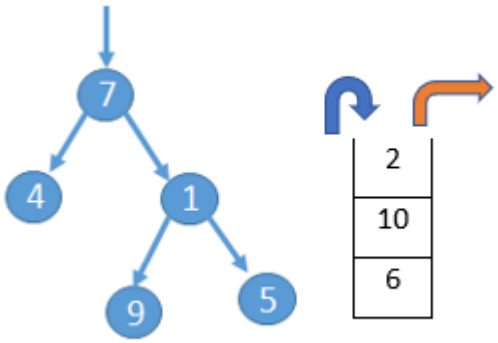
(6 נק') א. כתבו פעולה **bigger** המקבלת עץ בינארי **t** של מספרים שלמים ומספר שלם **x** ומחזירה את מספר הצמתים בעץ t שערכם גדולים מ- x.

(6 נק') ב. נתונה המחלקה **Item** הבאה:

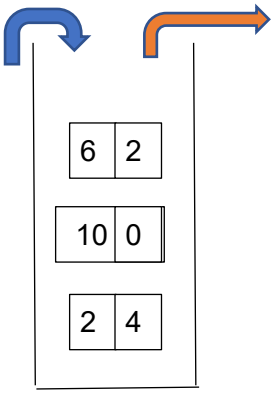
```
class Item
{
  private int num;
  private int bigNum;
  public Item(int n, int b)
  {
   this.num = n;
   this.bigNum = b;
  }
….
}
```

כתבו פעולה `bigInTree` המקבלת עץ בינארי `bt` ומחסנית של מספרים שלמים `s` ומחזירה מחסנית שבה כל איבר הוא מטיפוס `Item` שמכיל ערך ממחסנית `s` ומספר האיברים הגדולים ממנו שנמצאים בעץ `bt`.

לדוגמה: עבור עץ בינארי `bt` ומחסנית `s` הבאים:



הפעולה תחזיר את המחסנית הבאה:



(3 נק') ג. מהי הסיבוכיות של זמן הריצה של הפעולה? **הסבירו את תשובתכם**.

חלק ג׳

ענו על <u>שתיים</u> מבין השאלות 9-11 (ערך כל שאלה – 12 נקודות).

שאלה 9

במשטרת ירושלים שוטרים ממגוון התמחויות: חוקרים, חוקרי נוער, מפקדי התחנות.
מעוניינים למחשב את פרטי השוטרים ולשמור את פרטיהם באופן מסודר. לשם כך פותח פרויקט לניהול הצוות השוטרים. הפרויקט כולל ארבע המחלקות הבאות:

- Policeman (שוטר).
- Investigator (חוקר).
- SpecInvestigator (חוקר נוער).
- Commander (מפקד התחנה).

לחוקר **(Investigator)** יש את התכונות:

- name – שם, מטיפוס מחרוזת, String.
- id – מספר תעודת זהות, מטיפוס מחרוזת, String.
- seniority – ותק, מטיפוס שלם, int.
- eduLevel – רמת ההשכלה, מטיפוס מחרוזת, String.

לחוקר נוער **(SpecInvestigator)** יש את התכונות:

- name – שם, מטיפוס מחרוזת, String.
- id – מספר תעודת זהות, מטיפוס מחרוזת, String.
- seniority – ותק, מטיפוס שלם, int.
- eduLevel – רמת ההשכלה, מטיפוס מחרוזת, String.
- extensionStudy – עבר השתלמות מיוחדת, מטיפוס boolean. true – אם עבר, false – אם לא עבר.

מפקד התחנה **(Commander)** הוא **חוקר** שאחראי על הצוות שמונה עד 30 שוטרים.
למפקד תחנה יש את התכונות:

- name – שם, מטיפוס מחרוזת, String.
- id – מספר תעודת זהות, מטיפוס מחרוזת, String.
- seniority – ותק, מטיפוס שלם, int.
- eduLevel – רמת ההשכלה, מטיפוס מחרוזת, String.
- arr – מערך של שוטרים הכפופים למפקד התחנה.
- current – מספר אנשי הצוות בפועל, מטיפוס שלם, int.

(2 נק׳) א. שרטטו תרשים UML המתאר את הקשר בין המחלקות:
Policeman, Investigator, SpecInvestigator, Commander
באופן המתאים ביותר לעקרונות של תכנות מונחה עצמים.
יש לציין ייחסי ירושה והכלה.

(5 נק׳) ב. לכל אחת מהמחלקות **Policeman, Investigator, SpecInvestigator, Commander** כתבו:

- כותרת המחלקה.
- תכונות.
- פעולה בונה – הפעולה הבונה של כל מחלקה מקבלת את כל הפרמטרים הנדרשים.

(5 נק׳) ג. כתבו פעולה חיצונית המקבלת מערך שוטרים ומדפיסה את שמות מפקדי התחנות שבין השוטרים שבפיקודם יש חוקרי נוער אשר לא עברו השתלמות מיוחדת.
הערה: הניחו שהפעולות get ו- set מוגדרות בעבור כל תכונה בכל אחת מהמחלקות הפרויקט.

שאלה 10

במשרד הפנים החליטו להקים מאגר רחובות ושכונות בכל הערים בארץ. לשם כך פותח פרויקט הכולל שתי מחלקות עיקריות: City (עיר) ו- Quarter (שכונה).
המחלקה Quarter כוללת שלוש תכונות:

- name – שם שכונה, מטיפוס String.
- numPeople – מספר תושבים בתכונה, מטיפוס int.
- streetList – אוסף רחובות בשכונה.

המחלקה City כוללת שתי תכונות:

- name – שם עיר.
- quarterList – אוסף שכונות.

(2 נק׳) א. כתבו כותרות לשתי המחלקות, הגדירו את התכונות שלהן וכתבו פעולות בונות המקבלות שמות של שכונה או של עיר ומאתחלים אוסיפים להיות ריקים.
שימו לב: כדי לייצג אוסף אפשר להשתמש רק במבני נתונים הבאים:
מחסנית, תור, שרשרת חוליות.
אפשר להניח שבמחלקות Quarter ו- City קיימות פעולות set/get לכל תכונות.

(4 נק׳) ב. כתבו פעולה פנימית במחלקה City המקבלת שם רחוב street ומדפיסה שמות של כל השכונות שרחוב זה עובר בהן.
אפשר להניח שרחוב street קיים בעיר.

משרד הפנים קיבל החלטה לאחד שתי שכונות.

(6 נק׳) ג. כתבו פעולה במחלקה City המקבלת שמות של שתי שכונות ומבצעת את השינויים הנדרשים באוסף שכונות. השם של <u>השכונה</u> החדשה הוא שמה של השכונה שיש לה יותר תושבים.
שימו לב: אם בשתי תכונות יש אותו רחוב, אחרי האיחוד הוא צריך להופיע פעם אחת בלבד.

שאלה 11

נתונות ארבע המחלקות הבאות: אבא (Father), בן (Son) ונכד (GrandSon):

```
public class Father {

    public Father() {
    }
    public void smile() {
        System.out.println("Father is smiling");
    }
    public void run() {
        System.out.println("Father is running");
    }
}

public class Son extends Father {

    public Son() {
        super();
    }
    public void smile() {
        System.out.println("Son is smiling");
    }
    public void run() {
        System.out.println("Son is running");
    }
    public void work() {
        System.out.println("Son is working");
    }
}

public class GrandSon extends Son {

    public GrandSon() {
        super();
    }
    public void smile() {
        System.out.println("GrandSon is smiling");
    }
    public void run() {
        System.out.println("GrandSon is running");
    }
    public void play() {
        System.out.println("GrandSon is playing");
    }
}
```

(3 נק׳) א. עבור כל אחד משלושת קטעי הקוד הבאים, רשמו אם הוא תקין או לא. אם הקוד תקין – ציינו מה יהיה הפלט, אם הקוד אינו תקין – הסבירו מהו סוג השגיאה (שגיאת קומפילציה או שגיאת זמן ריצה).

1.
```
Father f = (Son)(new GrandSon());
```
2.
```
Object o = new Son();
Father f = o;
```
3.
```
Father f = new Father();
Object o = f;
GrandSon gs = (GrandSon)o;
```

(6 נק׳) ב. עבור כל אחד מחמשת קטעי הקוד הבאים רשמו אם הוא תקין או לא. אם הקוד תקין – ציינו מה יהיה הפלט, אם הקוד אינו תקין – הסבירו מהו סוג השגיאה (שגיאת קומפילציה או שגיאת זמן ריצה).

1.
```
Son s = new Father();
s.smile();
```
2.
```
Father f = new Son();
f.smile();
```
3.
```
Father f = new Son();
f.work();
```
4.
```
Son s = new GrandSon();
s.play();
```
5.
```
Father f = new GrandSon();
f.smile();
```

(3 נק׳) ג. עקבו אחרי קטע קוד הבא ורשמו מה יהיה הפלט.

```
Father[] f = new Father[3];
f[0] = new Father();
f[1] = new Son();
f[2] = new GrandSon();
for (int i = 0; i  <f.length; i++)
  if (f[i] instanceof Son)
     ((Son)f[i]).Run();
```

מבחן ב- C#

חלק א׳

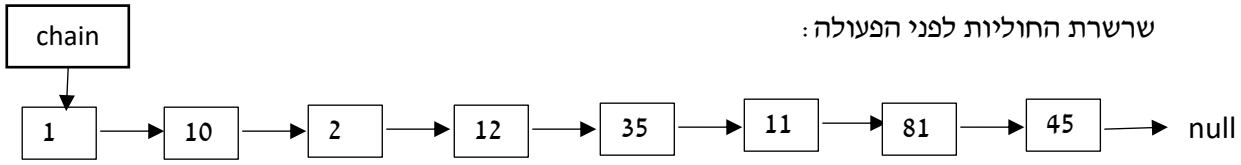
ענו על <u>שלוש</u> מבין השאלות 1-4 (ערך כל שאלה – 15 נקודות).

שאלה 1

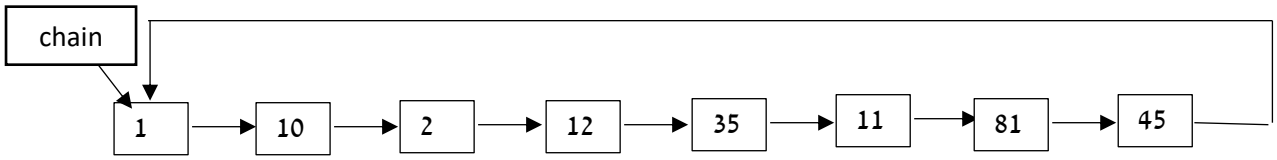
(8 נק׳) א. כתבו פעולה המקבלת הפניה לחוליה ראשונה בשרשרת חוליות של מספרים שלמים והופכת את השרשרת לשרשרת ״**מעגלית**״, כך שהחוליה אחרי החוליה האחרונה תהיה שוב החוליה הראשונה.

לדוגמה:

שרשרת החוליות לפני הפעולה:



שרשרת מעגלית אחרי הפעולה:



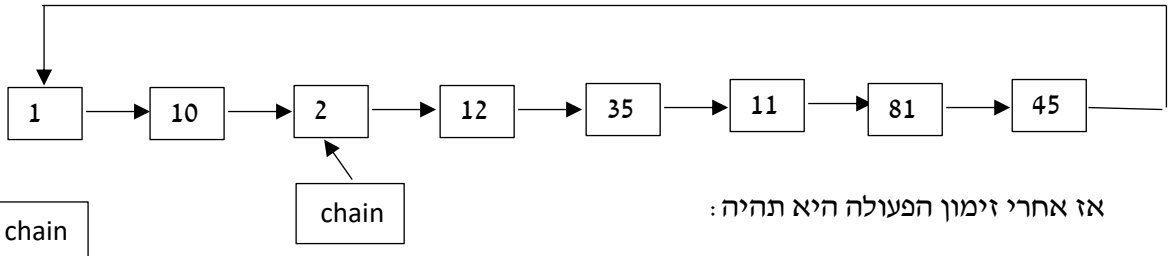
כותרת הפעולה:

```
public static void ToCircle(Node<Int> chain)
```

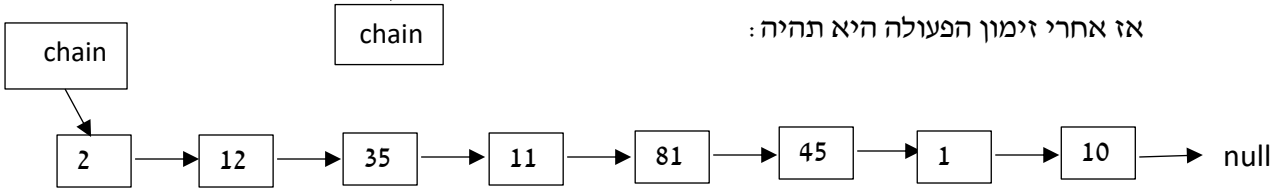
(7 נק׳) ב. כתבו פעולה המקבלת הפניה לחוליה כלשהי בשרשרת חוליות **מעגלית** של מספרים שלמים והופכת אותה לשרשרת לינארית, כך שהחוליה שהפנייה אליה התקבלה כפרמטר, תהיה חוליה ראשונה של השרשרת והחוליה שהייתה לפניה תהיה החוליה האחרונה בשרשרת.

לדוגמה:

אם לפני זימון הפעולה השרשרת הייתה:



אז אחרי זימון הפעולה היא תהיה:

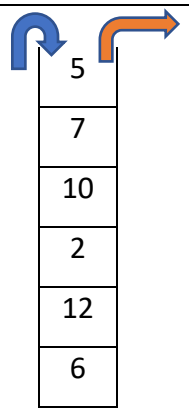
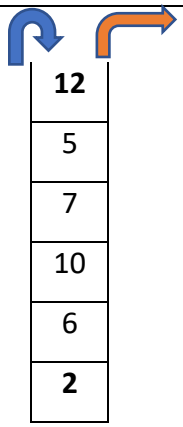


כותרת הפעולה: `public static void ToLine(Node<Int> chain)`

שאלה 2

(12 נק׳) א. כתבו פעולה המקבלת מחסנית של מספרים שלמים. הפעולה תעביר את הערך הקטן ביותר לתחתית המחסנית ואת הערך הגדול ביותר לראש המחסנית.
אפשר להניח שהערך הקטן ביותר וגם הערך הגדול ביותר מופיעים במחסנית פעם אחת בלבד.

לדוגמה:

המחסנית לפני הפעולה	**המחסנית אחרי הפעולה**
	

(3 נק׳) ב. מהי הסיבוכיות של הפעולה שכתבתם בסעיף א׳? הסבירו את תשובתכם.

שאלה 3

נתונות שתי המחלקות הבאות:

```
public class One  {
    private int num1=0;
    private int num2=0;
    protected static int count =0;

    public One(int num)    {
        this.num1 = num;
        this.num2 = num;
        count++;
        Console.WriteLine ("One ctor1");
    }

    public One(int num1, int num2)    {
        this.num1 = num1;
        this.num2 = num2;
        count++;
        Console.WriteLine ("One ctor2");
    }

    public virtual int Sum()    {return this.num1+this.num2;}
    public virtual void SetNum(int num)    { this.num2 = num;}
    public static int GetCount()    { return count;    }
}// end of class One

public class Two :One  {
    private int num3 = 0;

    public Two(int num):base(num)  {
        this.num3 = num;
        Console.WriteLine ("Two ctor1");
    }

    public Two(int num1, int num2, int num3): base(num1, num2) {

        this.num3 = num3;
        Console.WriteLine ("Two ctor2");
    }

    public override int Sum()  {return base.Sum()+this.num3; }
    public override void SetNum(int num)   {this.num3 = num; }
    public static int GetCount()  {return count;    }
}// end of class Two
```

נתונה המחלקה Driver הבאה:

```
public class Run  {
    public static void Main (string [] args)     {
            One f1 = new One(10);
            One f2 = new One (10, 20);
            Two s1 = new Two(1);
            One f3 = new Two(2);
(1)         Console.WriteLine ("count= " + One.GetCount());
            f2 = s1;
(2)         Console.WriteLine ("sum= " + f2.Sum());
            s1.SetNum(2);
(3)         Console.WriteLine ("sum= " + s1.Sum());
(4)         Console.WriteLine ("sum= " + f2.Sum());
            f2.SetNum(4);
(5)         Console.WriteLine ("sum= " + f2.Sum());
            f1.SetNum(4);
(6)         Console.WriteLine ("sum= " + f1.Sum());
(7)         Console.WriteLine ("count= " + Two.GetCount());
    }
}
```

מהו הפלט המתקבל לאחר הפעלת השיטה Main שבמחלקה Run?

כתבו את מספר השורה ואז את הפלט המתקבל משורה זו. אם יש שגיאת קומפילציה, כתבו אותה.

שאלה 4

נתונות שלוש מחלקות A ,B ,C הבאות:

```
public class A {
    public static int count = 0;
    private int num;
    public A()
    {
        count++;
        num = count;
    }
    public A(int n)
    {
        num = n;
    }
    public override string ToString()
    {
        return count+","+ num;
    }
} // end of A
public class B :A {
    private string str;
    public B()
    {
        str="GOOD";
    }
    public B(string s): base(10)
    {
        str = s;
    }
    public override string ToString()
    {
        return base.ToString()+" "+str;
    }
}// end of B
public class C : B {
    private string str;
    public C(string s)
    {
        str = s;
    }
    public override string ToString()
    {
        return base.ToString()+" "+str;
    }// end of C
```

(6 נק׳) א. נתונה מחלקה Driver הבאה:

```
public class Driver
{
    public static void Main(string[] args)
    {
        A[] arr=new A[6];

        arr[0]= … // (1)
        arr[1]= … // (2)
        arr[2]= … // (3)
        arr[3]= … // (4)
        arr[4]= … // (5)
        arr[5]= … // (6)


        for(int i=0; i<arr.Length; i++)
            Console.WriteLine(arr[i]);
    }
}
```

השלימו פעולות הסרות (1)-(6) כך שהפלט יהיה:

```
4,10 Morning

4,1

4,2 GOOD Afternoon

4,10 Evening

4,3 GOOD Night

4,4 GOOD
```

(6 נק׳) ב. כותרת המחלקה C השתנתה ל-

`public class C :A`

האם שינוי זה יגרום לשגיאה? אם כן – הסבירו את השגיאה, אם לא – רשמו מה יהיה הפלט של ה- Main אחרי השינוי.

(3 נק׳) ג. כותרת המחלקה C השתנתה ל-

`public class C`

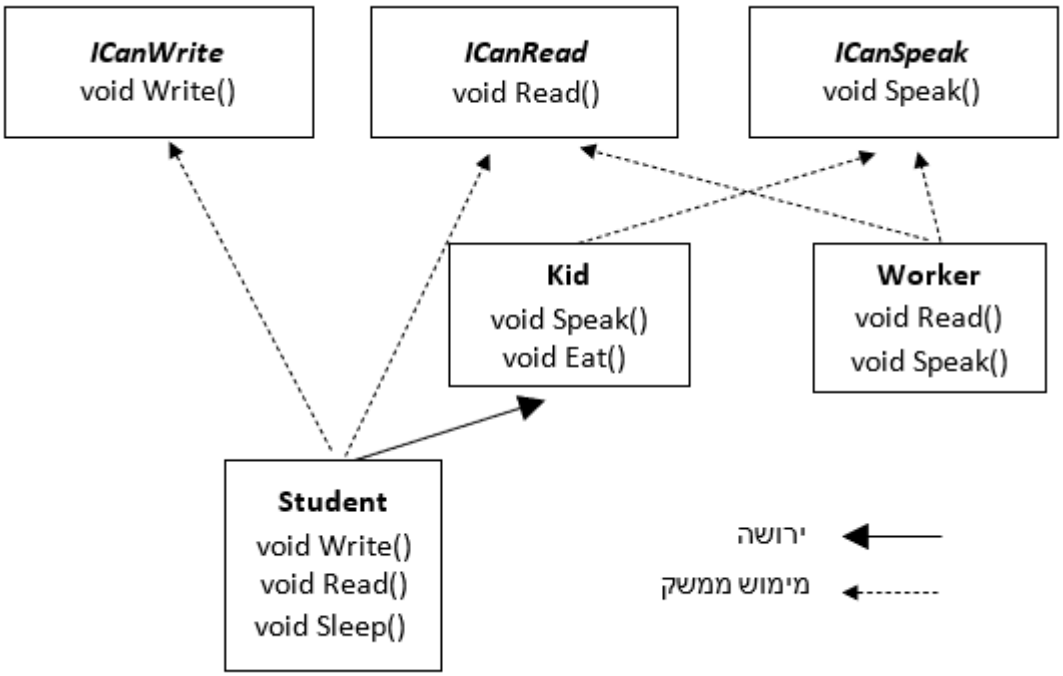
האם שינוי זה יגרום לשגיאה? אם כן – הסבירו את השגיאה, אם לא – רשמו מה יהיה הפלט של ה- Main אחרי השינוי.

חלק ב׳

ענו על <u>שתיים</u> מבין השאלות 5-8 (ערך כל שאלה – 15 נקודות).

שאלה 5

נתונה היררכיית המחלקות הבאה הכוללת שלוש מחלקות: Kid, Student, Worker ושלושה ממשקים: ICanRead, ICanWrite, ICanSpeak.



בכל מחלקה מוגדרת פעולה בונה ללא פרמטרים.

(3 נק׳) א. כתבו עבור כל מחלקה, את כותרת המחלקה.

(6 נק׳) ב. השלימו את שלוש שורות (1)-(3) כך שקטע קוד שלפניכם יעבור בלי שגיאות קומפילציה וזמן ריצה:

```
_____(1)_____   x = new Worker();
_____(2)_____   y = new Kid();
_____(3)_____   z = new Student();
x = y;
y = new Student();
y.Eat();
z.Read();
((Student)z).Speak();
((Kid)z).Eat();
z = new Worker();
```

(6 נק׳) ג. כתבו פעולה המקבלת כפרמטר מערך מטיפוס ICanSpeak. הפעולה תעבור על איברי המערך ותקרא לפעולה ()Write, אם היא קיימת עבור העצם. אם הפעולה ()Write אינה קיימת – תיקרא הפעולה ()Speak.

שאלה 6

שיטה אחת לדחיסת טקסט היא להחליף רצף של תווים זהים בתו אחד.

כדי לממש את השיטה הוגדרה המחלקה הבאה:

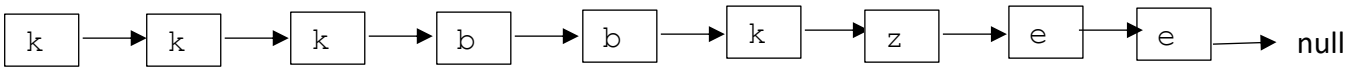
```
public class CharInt
{
    private char let;
    private int num;
    public CharInt(char let, int num)
    {
        this.let = let;
        this.num = num;
    }
  ….
}
```

תכונות המחלקה הן: תו (let) ומספר הפעמים שהוא מופיע ברצף (num). במחלקה הוגדרו פעולות Set/Get לכל תכונה וגם פעולה ToString.

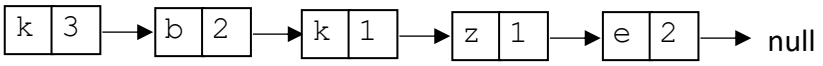
(9 נק׳) א. כתבו פעולה המקבלת הפניה לחוליה הראשונה של שרשרת חוליות של תווים (״טקסט מקורי״).
הפעולה מחזירה את הפניה לחוליה הראשונה של שרשרת חדשה של עצמים מטיפוס CharInt (״טקסט דחוס״).

לדוגמה:

אם שרשרת התווים היא:



אז הפעולה תחזיר את השרשרת הבאה:



כותרת הפעולה:

```
public static Node<CharInt> Zip(Node<char> chain)
```

(6 נק׳) ב. כתבו פעולה המקבלת הפניה לחוליה הראשונה של שרשרת המייצגת ״טקסט דחוס״ ומחזירה את הפניה לחוליה הראשונה של שרשרת תווים המייצגת ״טקסט מקורי״.

כותרת הפעולה:

```
public static Node<char> UnZip(Node<CharInt> chain)
```

שאלה 7

נגדיר טיפוס נתונים חדש בשם **DoubleQueue (תור כפול)**, כמבנה המכיל שני תורים של מספרים שלמים:

```
class DoubleQueue {
private Queue<int> first;
private Queue<int> second;
```

במחלקה הוגדרו ארבע הפעולות הבאות (**אין צורך לממש אותן**):

- הפעולה `public int Count (int queueID)` מחזירה מספר איברים בתור `queueID`.
- הפעולה `public void AddValue (int num, int queueID)` מוסיפה את הערך `num` לתור `queueID`.

(4 נק׳) א. כתבו פעולה המקבלת מספר תור `queueID` ומחזירה ערך הגדול ביותר בתור.

כותרת הפעולה:

```
public int MaxValue(int queueID)
```

(4 נק׳) ב. כתבו פעולה המקבלת מספר שלם `num` ומספר תור `queueID`.

כותרת הפעולה:

```
public void EraseValue(int num, int queueID)
```

הפעולה מוחקת מתור `queueID` את המספר `num`. אם המספר `num` מופיע יותר מפעם אחת בתור, הפעולה תמחק רק מופע אחד של המספר.

אם המספר `num` לא מופיע בתור, הפעולה אינה מבצעת דבר.

(7 נק׳) ג. כתבו את הפעולה

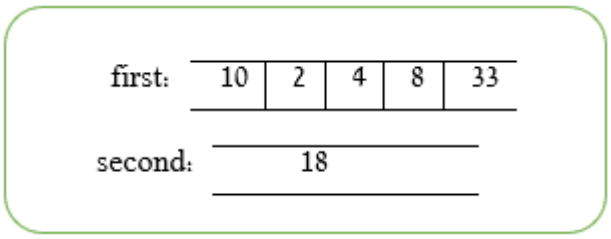
```
public static void Order (DoubleQueue dq)
```

הפעולה מקבלת תור כפול של מספרים שלמים dq. מספר האיברים בתור dq זוגי.

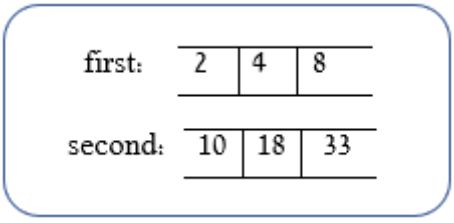
הפעולה תחלק את האיברים בין התורים first ו- second, לפי הכלל הבא:

- מספר האיברים בשני התורים זהה.
- ההפרש בין **סכום** האיברים הנמצאים בתור second ו**סכום** האיברים הנמצאים בתור first יהיה הגדול ביותר.

לדוגמה: תור כפול לפני זימון הפעולה order:



תור כפול אחרי זימון הפעולה order:



שימו לב! בסעיף ג׳ אפשר להשתמש רק בפעולות שהוגדרו במחלקה `DoubleQueue`.

אין להוסיף פעולות נוספות או להשתמש במבני נתונים נוספים!

שאלה 8

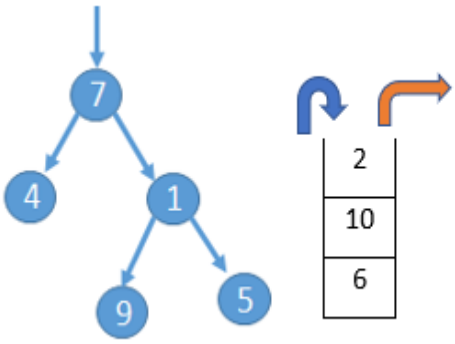
(6 נק') א. כתבו פעולה **Bigger** המקבלת עץ בינארי **t** של מספרים שלמים ומספר שלם **x** ומחזירה את מספר הצמתים בעץ t שערכם גדולים מ- x.

(6 נק') ב. נתונה המחלקה **Item** הבאה:

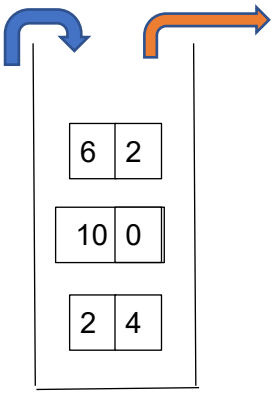
```
class Item
{
  private int num;
  private int bigNum;
  public Item(int n, int b)
  {
   this.num = n;
   this.bigNum = b;
  }
….
}
```

כתבו פעולה `BigInTree` המקבלת עץ בינארי `bt` ומחסנית של מספרים שלמים `s` ומחזירה מחסנית שבה כל איבר הוא מטיפוס `Item` שמכיל ערך ממחסנית `s` ומספר האיברים הגדולים ממנו שנמצאים בעץ `bt`.

לדוגמה: עבור עץ בינארי `bt` ומחסנית `s` הבאים:



הפעולה תחזיר את המחסנית הבאה:



(3 נק') ג. מהי הסיבוכיות של זמן הריצה של הפעולה? **הסבירו את תשובתכם**.

חלק ג׳

ענו על <u>שתיים</u> מבין השאלות 9-11 (ערך כל שאלה – 12 נקודות).

שאלה 9

במשטרת ירושלים יש שוטרים ממגוון התמחויות: חוקרים, חוקרי נוער, מפקדי תחנות.
מעוניינים למחשב את פרטי השוטרים ולשמור את פרטיהם באופן מסודר. לשם כך פותח פרויקט לניהול צוות השוטרים. הפרויקט כולל את ארבע המחלקות הבאות:

- Policeman (שוטר)
- Investigator (חוקר)
- SpecInvestigator (חוקר נוער)
- Commander (מפקד תחנה)

לחוקר (**Investigator**) יש את התכונות:

- name – שם, מטיפוס מחרוזת, string.
- id – מספר תעודת זהות, מטיפוס מחרוזת, string.
- seniority – ותק, מטיפוס שלם, int.
- eduLevel – רמת השכלה, מטיפוס מחרוזת, string.

לחוקר נוער (**SpecInvestigator**) יש את התכונות:

- name – שם, מטיפוס מחרוזת, string.
- id – מספר תעודת זהות, מטיפוס מחרוזת, string.
- seniority – ותק, מטיפוס שלם, int.
- eduLevel – רמת השכלה, מטיפוס מחרוזת, string.
- extensionStudy –עבר השתלמות מיוחדת, מטיפוס bool. true – אם עבר, false – אם לא עבר.

מפקד התחנה (**Commander**) הוא **חוקר** שאחראי על צוות שמונה עד 30 שוטרים.
למפקד תחנה יש את התכונות:

- name – שם, מטיפוס מחרוזת, string.
- id – מספר תעודת זהות, מטיפוס מחרוזת, string.
- seniority – ותק, מטיפוס שלם, int.
- eduLevel – רמת ההשכלה, מטיפוס מחרוזת, string.
- arr – מערך של שוטרים הכפופים למפקד התחנה.
- current – מספר אנשי הצוות בפועל, מטיפוס שלם, int.

(2 נק׳) א. שרטטו תרשים UML המתאר את הקשר בין המחלקות:

Policeman, Investigator, SpecInvestigator, Commander

באופן המתאים ביותר לעקרונות של תכנות מונחה עצמים.

יש לציין ייחסי ירושה והכלה.

(5 נק׳) ב. לכל אחת מהמחלקות **Policeman, Investigator, SpecInvestigator, Commander** כתבו:

- כותרת המחלקה.
- תכונות.
- פעולה בונה – הפעולה הבונה של כל מחלקה מקבלת את כל הפרמטרים הנדרשים.

(5 נק׳) ג. כתבו פעולה חיצונית המקבלת מערך שוטרים ומדפיסה את שמות מפקדי התחנות שבין השוטרים שבפיקודם יש חוקרי נוער אשר לא עברו השתלמות מיוחדת.

הערה: הניחו שהפעולות Get ו- Set מוגדרות בעבור כל תכונה בכל אחת מהמחלקות הפרויקט.

שאלה 10

במשרד הפנים החליטו להקים מאגר רחובות ושכונות בכל הערים בארץ. לשם כך פותח פרויקט הכולל שתי מחלקות עיקריות: City (עיר) ו- Quarter (שכונה).
המחלקה Quarter כוללת שלוש תכונות:

- name – שם שכונה, מטיפוס string.
- numPeople – מספר תושבים בתכונה, מטיפוס int.
- streetList – אוסף רחובות בשכונה.

המחלקה City כוללת שתי תכונות:

- name – שם עיר.
- quarterList – אוסף שכונות.

א. **(2 נק׳)** כתבו כותרות לשתי המחלקות, הגדירו את התכונות שלהן וכתבו פעולות בונות המקבלות שמות של שכונה או של עיר ומאתחלים אוסף להיות ריקים.
שימו לב: כדי לייצג אוסף אפשר להשתמש רק במבני הנתונים הבאים: מחסנית, תור, שרשרת חוליות.
אפשר להניח שבמחלקות Quarter ו- City קיימות פעולות Set/Get לכל תכונות.

ב. **(4 נק׳)** כתבו פעולה פנימית במחלקה City המקבלת שם רחוב street ומדפיסה שמות של כל השכונות שרחוב זה עובר בהן.
אפשר להניח שרחוב street קיים בעיר.

משרד הפנים קיבל החלטה לאחד שתי שכונות.

ג. **(6 נק׳)** כתבו פעולה במחלקה City המקבלת שמות של שתי שכונות ומבצעת את השינויים הנדרשים באוסף שכונות. השם של השכונה החדשה הוא שמה של השכונה שיש לה יותר תושבים.
שימו לב: אם בשתי תכונות יש אותו רחוב, אחרי האיחוד הוא צריך להופיע פעם אחת בלבד.

שאלה 11

נתונות ארבע המחלקות הבאות: אבא (Father), בן (Son) ונכד (GrandSon):

```
public class Father {

    public Father(){
    }
    public virtual void Smile() {
        Console.WriteLine("Father is smiling");
    }
    public virtual void Run() {
        Console.WriteLine("Father is running");
    }
}

public class Son :Father {

    public Son(): base(){
    }
    public override void Smile() {
        Console.WriteLine("Son is smiling");
    }
    public override void Run() {
        Console.WriteLine("Son is running");
    }
    public virtual void Work() {
        Console.WriteLine("Son is working");
    }
}

public class GrandSon :Son {

    public GrandSon():base() {
    }
    public override void Smile() {
        Console.WriteLine("GrandSon is smiling");
    }
    public override void Run() {
        Console.WriteLine("GrandSon is running");
    }
    public void Play() {
        Console.WriteLine("GrandSon is playing");
    }
}
```

(3 נק׳) א. עבור כל אחד משלושת קטעי הקוד הבאים, רשמו אם הוא תקין או לא. אם הקוד תקין – ציינו מה יהיה הפלט, אם הקוד אינו תקין – הסבירו מהו סוג השגיאה (שגיאת קומפילציה או שגיאת זמן ריצה).

```
1. Father f = (Son)(new GrandSon());
2. Object o = new Son();
   Father f = o;
3. Father f = new Father();
   Object o = f;
   GrandSon gs = (GrandSon)o;
```

(6 נק׳) ב. עבור כל אחד מחמשת קטעי הקוד הבאים רשמו אם הוא תקין או לא. אם הקוד תקין – ציינו מה יהיה הפלט, אם הקוד אינו תקין – הסבירו מהו סוג השגיאה (שגיאת קומפילציה או שגיאת זמן ריצה).

```
1. Son s = new Father();
   s.Smile();
2. Father f = new Son();
   f.Smile();
3. Father f = new Son();
   f.Work();
4. Son s = new GrandSon();
   s.Play();
5. Father f = new GrandSon();
f.Smile();
```

(3 נק׳) ג. עקבו אחרי קטע קוד הבא ורשמו מה יהיה הפלט.

```
Father[] f = new Father[3];
f[0] = new Father();
f[1] = new Son();
f[2] = new GrandSon();
for (int i = 0; i  <f.Length; i++)
  if (f[i] is Son)
     ((Son)f[i]).Run();
```

בהצלחה!

נספח לשאלון 97105 – מבני נתונים ותכנות מונחה עצמים – JAVA

נספח ממשקים מבנה הנתונים בתוכנית הלימודים

ממשק המחלקה חוליה הגנרית – Node<T>

המחלקה מגדירה חוליה גנרית שבה ערך מטיפוס T והפניה לחוליה העוקבת.

Node (T x)	הפעולה בונה חוליה. הערך של החוליה הוא x, ואין לה חוליה עוקבת.
Node (T x, Node<T> next)	הפעולה בונה חוליה. הערך של החוליה הוא x, והחוליה העוקבת לה היא next. ערכו של next יכול להיות **null**.
T GetValue()	הפעולה מחזירה את הערך של החוליה.
Node<T> GetNext()	הפעולה מחזירה את החוליה העוקבת. אם אין חוליה עוקבת, הפעולה מחזירה **null**.
void SetValue (T x)	הפעולה משנה את הערך השמור בחוליה ל- x.
bool HasNext()	הפעולה מחזירה **true** אם יש חוליה נוספת
void SetNext (Node<T> next)	הפעולה משנה את החוליה העוקבת ל- next. ערכו של next יכול להיות **null**.
override string ToString()	הפעולה מחזירה מחרוזת המתארת את החוליה.

יעילות הפעולות: כל הפעולות מתבצעות בסדר גודל קבוע, O(1).

ממשק המחלקה הגנרית – מחסנית Stack<T>

המחלקה מגדירה טיפוס אוסף בעל פרוטוקול **LIFO** להכנסה והוצאה של ערכים.

Stack()	הפעולה בונה מחסנית ריקה.
bool IsEmpty()	הפעולה מחזירה "אמת" אם המחסנית הנוכחית ריקה, "שקר" אם היא אינה ריקה.
void Push (T x)	הפעולה מכניסה את הערך x לראש המחסנית הנוכחית (דחיפה).
T Pop()	הפעולה מוציאה את הערך שבראש המחסנית הנוכחית ומחזירה אותו (שליפה). **הנחה:** המחסנית הנוכחית אינה ריקה.
T Top()	הפעולה מחזירה את הערך שבראש המחסנית הנוכחית בלי להוציאו. **הנחה:** המחסנית הנוכחית אינה ריקה.
override string ToString()	הפעולה מחזירה תיאור של המחסנית, כסדרה של ערכים, במבנה הזה (x_1 הוא האיבר שבראש המחסנית): $[x_1, x_2, \ldots, x_n]$

יעילות הפעולות – מחלקה מיוצגת בעזרת שרשרת חוליות.

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה () ToString המתבצעת בסדר גודל לינארי.

ממשק המחלקה הגנרית – תור <Queue<T>.

המחלקה מגדירה טיפוס אוסף עם פרוטוקול **FIFO** להכנסה והוצאה של ערכים.

הפעולה בונה תור ריק.	**Queue**()
הפעולה מחזירה "אמת" אם התור הנוכחי ריק, ו"שקר" אם הוא אינו ריק.	**bool** IsEmpty()
הפעולה מכניסה את הערך x לסוף התור הנוכחי	**void** Insert (Tx)
הפעולה מוציאה את הערך שבראש התור הנוכחי ומחזירה אותו. **הנחה:** התור הנוכחי אינו ריק.	**T** Remove()
הפעולה מחזירה את ערכו של האיבר שבראש התור מבלי להוציאו. **הנחה:** התור הנוכחי אינו ריק.	**T** Head()
הפעולה מחזירה מחרוזת המתארת את התור כסדרה של ערכים, במבנה הזה (x_1 הוא האיבר שבראש התור): $[x_1, x_2, \ldots, x_n]$	**override string** ToString()

יעילות הפעולות – המחלקה מיוצגת בעזרת שרשרת חוליות והפניה לזנב התור.

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה () ToString המתבצעת בסדר גודל לינארי.

נספח לשאלון 97105 – מבני נתונים ותכנות ונחה עצמים – C#

נספח ממשקים מבנה הנתונים בתוכנית הלימודים

ממשק המחלקה חוליה הגנרית – **Node<T>**

המחלקה מגדירה חוליה גנרית שבה ערך מטיפוס T והפניה לחוליה העוקבת.

הפעולה בונה חוליה. הערך של החוליה הוא x, ואין לה חוליה עוקבת.	**Node** (T x)
הפעולה בונה חוליה. הערך של החוליה הוא x, והחוליה העוקבת לה היא next. ערכו של next יכול להיות **null**.	Node (T x, Node<T> next)
הפעולה מחזירה את הערך של החוליה.	**T** GetValue()
הפעולה מחזירה את החוליה העוקבת. אם אין חוליה עוקבת, הפעולה מחזירה **null**.	**Node<T>** GetNext()
הפעולה משנה את הערך השמור בחוליה ל- x.	**void** SetValue (T x)
הפעולה מחזירה **true** אם יש חוליה נוספת	**bool** HasNext()
הפעולה משנה את החוליה העוקבת ל-next. ערכו של next יכול להיות **null**.	**void** SetNext (Node<T> next)
הפעולה מחזירה מחרוזת המתארת את החוליה.	**override string** ToString()

יעילות הפעולות: כל הפעולות מתבצעות בסדר גודל קבוע, O(1).

ממשק המחלקה הגנרית – מחסנית Stack<T>

המחלקה מגדירה טיפוס אוסף בעל פרוטוקול **LIFO** להכנסה והוצאה של ערכים.

הפעולה בונה מחסנית ריקה.	**Stack**()
הפעולה מחזירה "אמת" אם המחסנית הנוכחית ריקה, "שקר" אם היא אינה ריקה.	**bool** IsEmpty()
הפעולה מכניסה את הערך x לראש המחסנית הנוכחית (דחיפה).	**void** Push (T x)
הפעולה מוציאה את הערך שבראש המחסנית הנוכחית ומחזירה אותו (שליפה). **הנחה**: המחסנית הנוכחית אינה ריקה.	**T** Pop()
הפעולה מחזירה את הערך שבראש המחסנית הנוכחית בלי להוציאו. **הנחה**: המחסנית הנוכחית אינה ריקה.	**T** Top()
הפעולה מחזירה תיאור של המחסנית, כסדרה של ערכים, במבנה הזה (x_1 הוא האיבר שבראש המחסנית): $[x_1, x_2, \ldots, x_n]$	**override string** ToString()

יעילות הפעולות – מחלקה מיוצגת בעזרת שרשרת חוליות.

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה () ToString המתבצעת בסדר גודל לינארי.

ממשק המחלקה הגנרית – תור Queue<T>.

המחלקה מגדירה טיפוס אוסף עם פרוטוקול **FIFO** להכנסה והוצאה של ערכים.

הפעולה בונה תור ריק	**Queue**()
הפעולה מחזירה "אמת" אם התור הנוכחי ריק, ו"שקר" אם הוא אינו ריק	**bool** IsEmpty()
הפעולה מכניסה את הערך x לסוף התור הנוכחי	**void** Insert (Tx)
הפעולה מוציאה את הערך שבראש התור הנוכחי ומחזירה אותו. **הנחה**: התור הנוכחי אינו ריק	**T** Remove()
הפעולה מחזירה את ערכו של האיבר שבראש התור מבלי להוציאו. **הנחה**: התור הנוכחי אינו ריק	**T** Head()
הפעולה מחזירה מחרוזת המתארת את התור כסדרה של ערכים, במבנה הזה (x_1 הוא האיבר שבראש התור): $[x_1, x_2, \ldots, x_n]$	**override string** ToString()

יעילות הפעולות - המחלקה מיוצגת בעזרת שרשרת חוליות והפניה לזנב התור.

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה () ToString המתבצעת בסדר גודל לינארי.

שאלון 97105 מבני נתונים ותכנות מונחה עצמיים – מועד ב' אביב 2022

שאלה	סעיף	תת-סעיף	ניקוד	הערות
1	א	-	8	חריגה – להוריד 2 נק'. לולאה אין סופית – להוריד 2 נק'.
	ב	-	7	חריגה – להוריד 2 נק'. לולאה אין סופית – להוריד 2 נק'.
2	א	-	12	הערה: אין חשיבות לסדר איברים בתוך מחסנית.
	ב		3	בלי הסבר – לא לתת נק'.
3	א	-	15	• כל סעיף – 2 נק'.
4	א	-	6	• כל סעיף – 1 נק'.
	ב	-	6	
	ג		3	• כל סעיף – 1 נק'.
5	א	-	3	אם לא ציין ממשק – להוריד 0.5 נק' עבור כל מחלקה.
	ב		6	• כל סעיף – 2 נק'.
	ג	-	6	• כותרת פעולה – 1 נק'. • מעבר – 1 נק'. • בדיקה והמרה – 1 נק'. • מניה והחזרת ערך – 1 נק'.
6	א	-	9	חריגה – להוריד 2 נק'. לולאה אין סופית – להוריד 2 נק'.
	ב	-	6	חריגה – להוריד 2 נק'. לולאה אין סופית – להוריד 2 נק'. אם שגה במספר הופעות ב- 1 – להוריד 1 נק'.
7	א	-	4	אם לא שמר על התור – להוריד 2 נק'.
	ב	-	4	אם מחק את כל המופעים של מספר – להוריד 2 נק'.
	ג		7	אם השתמש בפעולות אחרות – להוריד 3 נק'.
8	א	-	6	
	ב	-	6	אם לא התשמש בפעולה של סעיף א' – להוריד 2 נק'.
	ג	-	3	בלי הסבר – לא לתת נקודות. ההסבר חייב להתייחס גם למספר צמתים בתור וגם למספר איברים במחסנית.
9	א	--	2	אם הפך כיוון חצים – להוריד 1 נק'.
	ב		5	אם לא ציין יחסי ירושה – להוריד 2 נק'.

				אם העתיק תכונות בכל מחלקה – להוריד 2 נק'.
	ג	-	5	
10	א	-	2	
	ב	-	4	אם השתמש במחסנית/תור ולא שמר – להוריד 1 נק'.
	ג		6	
11	א	-	3	כל סעיף – 1 נק'.
	ב	- -	6	כל סעיף – 1 נק'.
	ג		3	כל הדפסה – 1 נק'.

דחוף!

לכבוד
המכללות ובתי הספר
להכשרת הנדסאים וטכנאים

הנדון: הבהרה לבחינת גמר ממלכתית

תאריך בחינה: 27.3.2022		שעת העברה בדוא"ל:	10:59
מגמה:	**הנדסת תוכנה**		
שם הבחינה:	מבני נתונים ותכנות מונחה עצמים		
סמל הבחינה	97105		

<u>הבהרה:</u>

<u>1) עמ' 5 (JAVA) עמ' 20(C#)</u>

<u>במקום "נתונה מחלקה DRIVER" יש לכתוב "נתונה מחלקה"</u>

<u>2) עמ' 10(JAVA) עמ' 25 (C#)</u>

<u>בשאלה 7 יש להתייחס לפרמטר queueID כמספר תור :</u>

<u>queueID=1 - מציין תור FIRST</u>

<u>queueID=2 - מציין תור SECOND</u>

בברכה,
מחלקת בחינות

01-3-07 (01/07)